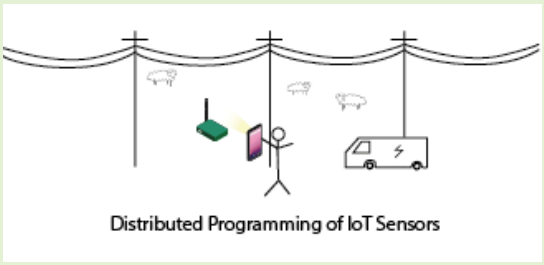
Secure Decentralised Deployment of LoRaWAN Sensors

Ross McPherson, and James Irvine, *Senior Member, IEEE*

***Abstract*— Low-power wide-area networks (LPWAN) technologies, such as LoRaWAN, have become a popular and cost-effective way of monitoring assets. Two considerations which still present a barrier to deployment are the cost of deployment and the potential cost and disruption of re-keying a compromised network. This loss of functionality from a compromised network has made security conscious industries reluctant to embrace LPWAN technology. This paper will address these concerns by simplifying the deployment and re-keying of LoRaWAN devices, by detailing a procedure which uses a smartphone's camera flash to transfer the necessary credentials. Smartphones were chosen as a transfer mechanism since they are both abundant and suitably powerful to generate and transfer secure keys. Using smartphones and light also removes the need for a laptop, a wired connection and programming software, allowing devices to be provisioned out in the field without the need for calibration or specialised tools. The design was created and successfully programs sensor devices in variety of environments, and has demonstrated benefits to critical national infrastructure industries such as utilities.**



***Index Terms*— Internet of Things, Security, LoRaWAN, Consumer Devices, Smart Grid**

I. Introduction

LORAWAN is a low-power wide-area network (LPWAN) which operates in the 868 MHz license-free frequency band and has become a popular choice for organisations wanting to deploy a cost-effective wireless sensor network. It benefits from an extensive ecosystem and considerable security research to find and resolve any potential flaws [1]. The devices themselves secure communication using symmetric cryptography, which requires both parties to know a shared key before any secure communication can commence. Therefore, a secure key-exchange must take place so that both parties have the same key. This creates two issues: (1) a suitably random key must be generated. (2) The key must be securely transferred to both the LoRaWAN device and LoRaWAN application server. Standalone LoRaWAN devices cannot generate suitably random keys [2], instead the device installer must generate a key to be installed into the device. This creates a barrier for organisations which either do not have this capability or knowledge. Security organisation IO Active recently detailed that LoRaWAN networks, in particular, are susceptible to insecure deployments due to a lack of understanding from organisations deploying the these networks [3]. In their findings they detailed that as LoRaWAN is advertised as secure by default, consumers believe no customization is required, thereby using default or easily guessable credentials.

The second issue is key transfer. High powered devices such as smartphones or personal computers avoid these problems by using public key infrastructure (PKI) to setup a secure channel, which is then used to transfer a shared secret [4]. This shared secret is then used to secure all messages between the device and the server. Although it is possible for LPWAN networks to support PKI, it is uncommon with only one commercial offering, due to the battery consumption and resource utilisation of such an operation. Therefore, the device installer must securely transfer the generated key to the LoRaWAN device, and to the corresponding application server.

Members of the LoRa alliance have attempted to resolve this dependency by creating pre-programmed modules with matching credentials which can be transferred to the new owner. However, this still presents two issues. The third party must be trusted to correctly generate, store, transfer, and subsequently delete the keys once they had been issued to the device owner. Security-conscious organisations such as critical national infrastructure (CNI) are held accountable for their suppliers [5]. Therefore, without oversight into how a third party programs their devices, CNI organisations would be unwise to select this option. The second issue is that if the network was compromised, the devices would have to be recalled and sent back to the manufacturer [6], a costly operation in terms of finances, labour, and loss of functionality.

One such CNI is the electricity distribution network. Although telemetry is currently fairly limited, with the introduction of smart grid this will rapidly expand. As such, LoRaWAN is being investigated in this role by members of the University of Strathclyde's Power Network Demonstration Centre (PNDC), a smart grid testing facility [7]. Even if

Both authors are with the University of Strathclyde, Electrical Engineering Department, Glasgow G1 1XQ (e-mail: ross.mcpherson@strath.ac.uk).

LoRaWAN devices are used only for measurement and not directly for control, if sensor networks are compromised the resulting readings will cause the network management to make false decisions. High-quality security in such networks is therefore essential, even when the distributed nature of such networks makes deployment difficult and costly. Moreover, electrical networks must be highly reliable and a loss of functionality would be highly detrimental and be a laborious process to replace each of the thousands of deployed devices for re-keying. Therefore, this paper will investigate a way to simplify installing shared keys into low-cost LoRaWAN devices, without relying on third parties.

II. Background

Using the recommended Over-The-Air-Activation (OTAA) procedure LoRaWAN 1.0, currently the most widely deployed version [1], requires an Application Key (AppKey), an Application Identification (AppEUI) and a Unique Device Identifier (DevEUI) to be present in both the application server and the device before the device can join the network [1]. This AppKey is used as a seed to generate an Application Session Key (AppSKey), and a Network Session Key (NwkSKey). The application layer uses the AppSKey to encrypt the payload of the message and ensuring data privacy, while the NwkSKey is responsible for authenticating messages [**?**]. Given all encryption and authentication in the network is derived from the AppKey, the security of communication is dependent on how unpredictable these keys are. If the keys were generated in any deterministic way the whole network could be compromised. This creates a requirement for random keys, which is a common problem in the field of computer security, normally keys are generated, using random data sources, such as the inherently random nature of human interaction with computers [8] or non-deterministic sources. A popularised example of this is a video recording a wall of lava lamps, which creates an entirely random image [9].

The challenge with low-cost embedded devices, such as LoRaWAN devices, is that they generally have a single purpose, and little variety in their scheduled tasks to generate randomness. They also lack human interaction and by default have no built-in hardware for random number generation. The original LoRaWAN specification designers required some random data for the generation of the message authentication code, but without the capability to create random numbers by default, they decided to use the received signal strength as a pseudorandom number generator. However, researchers were able to influence the received signal strength, and make a fixed nonce [10]. This demonstrates the complexity of generating secure random numbers, where even apparently random data can be influenced.

Various researchers have acknowledged these limitations of low-cost LPWAN devices, and have proposed several schemes to generate session keys in alternative ways [11]. Han et.al, noted that current scheme uses AES electronic codebook (ECB) to generate session keys from the root AppKey and through using pattern analysis the static AppKey, could be determined [12]. Consequently, they, proposed a new lightweight key update scheme to routinely update the AppKey, using a scheme resistant to this attack [12]. Zhang et.al, also detailed the challenges of generating secure and unique session keys on LoRaWAN devices, but instead proposed a new architecture, where a trusted third party generates the required session keys [13]. This third party will encrypt the keys using the pre-existing AppKey, and then transfer the encrypted keys to the LoRaWAN end device and network server. However, this also relies on a pre-shared key to encrypt the session keys while in transit. However, both methods required the installation of a pre-shared key on the device and the application server.

This reliance of a pre-existing key for low-cost devices is a common one, Xu et.al, noted as these devices do not support public key infrastructure they are dependent on a pre-installed key [14]. In response a new research field has emerged, physical layer key generation, where undetectable information is shared between the communicating parties to generate a shared key [13], [14]. Zhang et.al, and Xu et.al, proposed using LoRaWAN's received signal strength parameter as shared knowledge between the gateway and the device [13], [14]. Both sets of results show they were able to successfully generate a key between the end device and the gateway. However, as the key is generated on gateway and the end device, this invalidates the end-to-end encryption model of LoRaWAN, making the gateway a target for potential attackers. Xu et.al, also noted that an adversary listening to this information exchange could impersonate a legitimate actor and disrupt the key exchange protocol [14]. Consequently, Xu proposed including a message authentication code, which would require a pre-shared key. Therefore, each proposal requires a pre-shared key. Adding additional hardware to a LoRaWAN device, such as a physically unclonable function, or a hardware security module would allow for random number generation, and therefore key generation and transfer. However, this creates a barrier requiring additional hardware which would increase the cost of the device. IoT devices are price sensitive, and even a small increase in cost deters consumers [15], [16]. Manufacturers also struggle to differentiate themselves based on security features so instead invest in more marketable features [17]. One industry which is particularly price sensitive is the utility industry.

The regulated nature of utilities makes it challenging to invest in security, as expenditure has to be pre-allocated in eight year blocks, and difficulties in measuring the outcomes of security projects [18], [19], means that security is often only added on as sub-goal of larger projects [20]. Regulatory bodies are also often unwilling to allow for increased costs for security, arguing that they expect the networks to be secure by default. This creates strict financial constraints for utilities to spend on security, meaning that little additional cost could be added to a device, particularly as this cost would be replicated many times for a large deployment. Another constraint for utility companies is that devices are expected to be operational for decades, while advances in quantum computing may render most existing implementations of public key cryptography unusable, while systematic systems are still regarded as quantum safe [21].

III. Concept

To reduce the cost of a LoRaWAN deployment while delivering high security, a system is proposed which uses a smartphone as the basis for deployment. The smartphone will generate all the required security credentials for the LoRaWAN device and transfer them to the device and the application server. Smartphones were chosen as they are widely available, powerful, understandable, and in general more robust than laptops [22]. Moreover, they are capable of setting up secure TLS connections so that the generated credentials are securely transferred to the LoRaWAN application server. Transferring the credentials to the device itself is more challenging. Smartphones are fitted with an array of interfaces such as Bluetooth, Wi-Fi and NFC. However, by default, these devices do not feature on LoRaWAN sensor devices. Therefore additional receiving hardware would be required to receive the credentials using any of these interfaces, incurring additional manufacturing cost. Moreover, new radio devices must be tested and certified before they can be advertised and sold on the market [23]; a costly and time-intensive process. Instead, using a smartphone's LED camera flash is proposed. Messages are encoded using the flash and received by a light sensor on the LoRaWAN device, using a morse code-esque system.

One issue with this proposal is the communication is unidirectional, where the LoRaWAN device can only receive data. Hence, the device cannot respond once the credentials entry is complete. Although some sensor devices have status LEDs or similar, which could be used to signal the receipt of the credentials, many do not. Instead, the mobile application subscribes to the LoRaWAN application server, to be alerted once the device has successfully joined the LoRaWAN network, completing the process. The result is that the only additional hardware required for provisioning is a very low-cost light sensor both in terms of financial and energy consumption.

Consequently, the proposed solution is described as follows. The smartphone first receives the AppEUI from the deployment organisation's server over HTTP. This can be done in advance of an actual deployment as these credentials are not device-specific. Next, the smartphone generates the AppKey and transfers both credentials to the sensor device, along with any other required credentials. The smartphone then sets up a secure session to the LoRaWAN application server using TLS, to transfer the matching credentials to the server. Finally, the LoRaWAN device repeatedly sends a join request to the LoRaWAN application server and, upon successful connection, the smartphone will be alerted that the process is complete over cellular internet and will destroy the local copy of the unique credentials.

This is detailed in Fig. 1 along with the following list of bullet-points:

1) The smartphone assembles required LoRaWAN credentials, such as AppEUI.
2) The smartphone transfers credentials to the LoRaWAN device.
3) The smartphone transfers the credentials to the application server from the smartphone of a cellular or Wi-Fi connection.

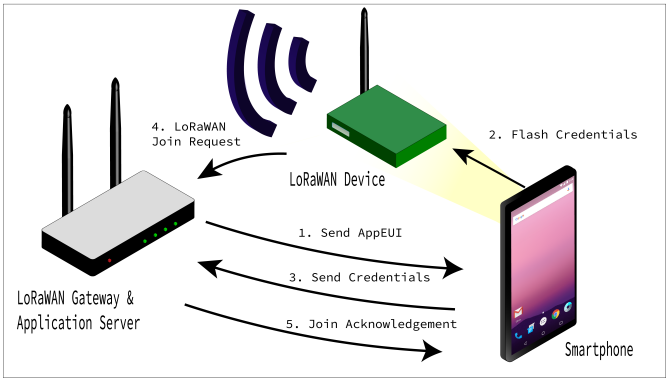


Fig. 1. LoRaWAN Device Setup Procedure

4) The device initiates a LoRaWAN join request using the credentials.
5) The smartphone receives an acknowledgement that the device is successfully connected.

IV. Key Transfer

While most smartphones have an LED camera flash, it is not intended for transmitting information, so some innovation in implementation is required. The LED can either be on or off. Three of the most prominent, yet straightforward transmission protocols, are On-Off Keying (OOK), Pulse Position Modulation (PPM), and Pulse-width modulation (PWM) [24], [25]. OOK is the simplest transmission method, where the presence of a pulse represents one value (e.g., '1') and the absence of a pulse the other (e.g., '0'). PMM, on the other hand, uses the position within a particular time slot to determine the received value. The downside with this approach is that the transmitter and the receiver both have to be synchronised so that the position within the time slot is known. Finally, PWM uses the duration of the pulse to determine the received value. As the very accurate timing of the pulse was impossible to determine, for reasons discussed below PWM was selected.

Possible light sensors are a light-dependent resistor (LDR) and a phototransistor. Phototransistors are generally cheaper than LDRs and have a faster response. To confirm these predicted characteristics, an Android application was created to periodically toggle the flashlight every 100 ms, and the phone was directed towards the sensor. The LDR took 40 ms to fully recognise the change in light level, while the phototransistor only required 10 µs. Given the faster response time and lower cost of the phototransistor, a phototransistor was used for the prototype.

The operation of the LED flash on Android smartphones is controlled through the operating system and is not directly visible to an installed app, which causes issues with accurate timing. The first phone tested, a OnePlus 5T, was configured to have the LED on for 10 ms, but the LED would stay illuminated for 16 ms. This prolonged light was fairly constant, with similar results with the LED being set on for 20, 30 and 50 ms, each resulting in approximately an additional six ms of recorded light. The app was then moved to to various different phone models, and each LED response time was measured, as shown in Table I, they all overran by four, five or six ms.

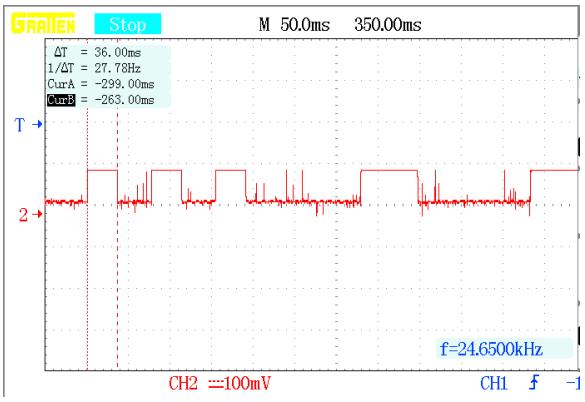


Fig. 2. Phototransistor Light Detection from 30 ms LED flash

Three further tests were undertaken to confirm that this behaviour is predictable in all conditions: (1) all other applications on the phone were closed; (2) assorted common applications were run; and (3) the periodic sequence of the light being on and off was replaced with a randomly generated sequences of on and off pulses, mimicking a randomly generated AES key. In each case, each illumination was extended by 4-6 ms. Fig. 2 shows the LED response from the OnePlus 5T, when tasked with flashing an aperiodic sequence, which shows that the width of the recorded pulse was 36 ms, while the programmed time for a pulse was 30 ms.

While relatively consistent at approximately 6 ms, this delay was a combination of the response time of the smartphone's camera hardware abstraction layer, the Android scheduler, and how many tasks the phone was undertaking. As the proposed system was designed to be applicable to a range of smartphones there were to many variables to rely on this. Therefore, very accurate timing is not be possible, which makes OOK and PPM unsuitable; consequently, low rate PWM was implemented. As two symbols had to be encoded, each one with a different pulse length, one-third of the transmission time was devoted to representing a zero, and two-thirds of the transmission period representing a one. The total transmission time would have to be suitably larger than the largest overrun measured, so the overrun would not change the measured result. A symbol period of 50 ms was chosen, allowing a transmission rate of 20 bit/s, making each third of a pulse 16.7 ms long. This creates a difference in symbols significantly larger than the expected overrun of 6 ms, so that a symbol could not overrun into the next threshold and be misinterpreted. Fig. 3 shows the two-thirds pulse has a 17 ms buffer before it would run into the next period. Similarly, the one-third pulse has a 16 ms buffer before it could be considered a long pulse. Both buffers are more than double the highest recorded overrun.

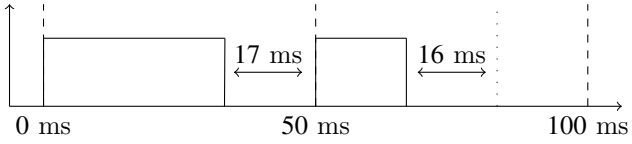


Fig. 3. Pulse Width Buffer for Overruns

The Android app was originally programmed to loop through the message one bit at a time, switch the light to the corresponding state and then sleep for the duration of the pulse. However, as the Android system call waited until the response light was off to return, the overrun time being added to each system call. This meant that instead of the first symbol ending at 50 ms, it would end at 56 ms. Therefore, by the completion of the tenth symbol the elapsed time was 560 ms. Consequently, the receiver expected 11 symbols to have been transmitted within this time, while only ten would have been sent, creating an inconsistency in the sent and received signals, and as noted this discrepancy would be variable depending on the hardware. A potential solution would be to extend transmission time of an individual message until the receiver had enough redundant data to discard any mismatch, but this would increase the overall transmission time. The more effective solution was to modify the Android application to pre-schedule the toggle instructions so that even if a small error was introduced, it would not cascade into all of the symbols, rather it would be rectified on the next symbol. Employing such a revised system would make the first pulse end at $t = 50$, the second at $t = 100$, and the tenth at $t = 500$.

Although Android has an Alarm Manager built into the Android Software Development Kit (SDK) specifically for periodic tasks, this library is unsuitable for sub-second precision, as Google opted to let the operating system slightly modify the pre-scheduled trigger time to optimise battery life better [26]. Instead, an array of Runnable tasks was created, where each task is staggered to set the corresponding light value based on the position in the message. This modified configuration resolves the issue with the first few samples triggering at the expected time, but a slight linear delay was introduced after the first flash. This was due to the time taken to execute the loop, causing each subsequent iteration to be set slightly after the correct time. To resolve this issue, the algorithm was modified to subtract the time taken to execute the loop as shown in Equation 1.

$$delay = (50 \times (1 + index)) - (start_{time} - loop_{time}) + 4 \quad (1)$$

V. Implementation

To test the light key transfer solution, a Printed Circuit Board (PCB) featuring a phototransistor, a MSP430FR6989 microcontroller and a LoRaWAN module was created Fig. 4. On start-up, the MSP430 was programmed to routinely sample the phototransistor converting the detected value to a digital representation through the built-in analogue to digital converter. These samples were stored and once the rolling average of the 20 previous symbols was deterministically higher than the average of the 50 prior symbols, this would signify the message preamble has been detected and initiate recording of all subsequent samples. Configuring the microcontroller to sample the phototransistor every millisecond meant that for each transmission period 50 samples would be taken. Through analysis of each of these groups of samples, a long or a short pulse would be determined.Once the user positions their smartphone camera flash over the phototransistor and initiates

Phone Model	Set Light (ms)	Actual Light (ms)	Set Light (ms)	Actual Light (ms)	Set Light (ms)	Actual Light (ms)
OnePlus 5T	50	56	30	36	10	16
Google Pixel 3	50	55	30	35	10	14
Nexus 6P	50	54	30	34	10	14
Sony Xperia XZ2	50	54	30	34	10	14
Motorola Moto G4	50	55	30	35	10	15

TABLE I
PHONES TESTED FOR LED RESPONSE TIME

the process, the app creates a binary string containing the preamble, credentials, and cyclic redundancy check (CRC). First, one second worth of preamble is added to the string, then an Appkey is securely generated using well-established crypto APIs [27]. The AppKey and previously received AppEUI are converted to a binary format, and a 16-bit CRC applied [28]. This result is then appended to the preamble string to create the overall message string. The app then schedules a series of delayed tasks, each staggered by 50 ms, to light the camera flash either for a long or short pulse, depending on the corresponding bit in the message. Concurrently, the Android app also initiates a TLS connection to register the device on the LoRaWAN application server using the phone's cellular connection to transfer the matching credentials, and upon successful transfer deletes the local copy.

The firmware on the sensor device would know what data it is expecting from the flashing light. Therefore, after the initial preamble message, the sensor device knows the message length without requiring synchronisation. Consequently, after the expected period has elapsed, the received CRC is validated against the generated one. If the message is successfully validated, the credentials will be extracted; otherwise, the sensor device will reset and continue waiting to be configured. Using the received credentials a LoRaWAN join request will be attempted. If successful, the application server issues an acknowledgement to the smartphone. If no such acknowledgement is received, potentially due to credentials mismatch in the join request, the app will reset and start flashing the same message again, until a successful acknowledgement is received or the programming is cancelled. The receiving device will validate the CRC for each received mesage, and if the CRC is validated the credentials are saved otherwise the device continues listening. Once the message is complete the companion app may also automatically detect additional information about the installation, and feed it back to the application server, for example, the GPS location of the installed site.

VI. ANALYSIS

During a full system test, the created LoRaWAN sensor device was provisioned in 17.1 seconds, without the need for specialist tools or knowledge on the part of the phone user. This was broken down into 1 second for the preamble, 12.8 seconds for transfer of AppKey, AppEUI, and DevEUI, 0.8 seconds for transfer of CRC check, and 2.4 seconds for saving the credentials to the microcontroller, checking

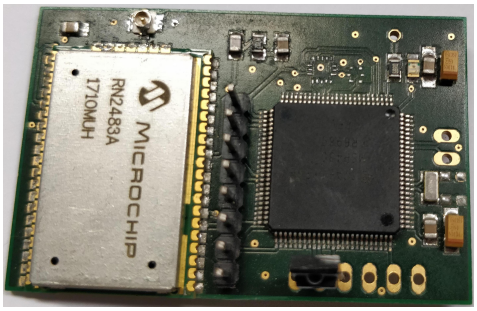


Fig. 4. LoRaWAN Sensor Board Featuring an LED Diode for Key Transfer

Task	Time (seconds)
Preamble	1
Key Transfer	12.8
CRC Transfer	0.8
MCU Processing	2.4
Total	**17.1**

TABLE II
TRANSFER AND PROCESSING TIME OF SENDING CREDENTIALS FROM SMARTPHONE TO SENSOR DEVICE

the CRC, joining the LoRaWAN network, and issuing a confirmation message to the Android handset, and is shown in Table II. LoRaWAN 1.0 was used for the implementation, but LoRaWAN 1.1 would be compatible with a slightly longer programming time due to the addition of the Network Server Key.

The programming application, which was ran an OnePlus 5T, required minimal CPU resources (approximately 1% of CPU), and the whole process took 956,416 us of processor time, with CRC generation and AES key generation taking 12,452, and 1,237 us respectively. The application also only required 80mb of RAM to operate, with 8.2 MB of code, 31 MB of graphics, and 30.4 MB of native C/C++ code used to interact with the underlying hardware. This meant that the resource requirement was low, and the majority of modern phones would be able to run the application. Another area of consideration is the energy usage. Although the energy impact of running the application itself was minimal, it requires ≈ 1.7 mAh of energy to illuminate the LED [29] depending on the exact number of ones and zeros in the message. Comparing this energy usage to the capacity of modern mobile phone

batteries, would allow for an estimated 1000 devices to be programmed before the phone would need to be charged.

The system was implemented using LoRaWAN 1.0, as this is the most common implementation at the time of writing [1]. However, as the proposal can transfer arbitrary data, the setup procedure can be modified with a software update to improve device longevity. This could involve using LoRaWAN 1.1 or adding additional security keys to the microcontroller itself. For example, although LoRaWAN only natively supports AES 128, the MSP430FR6989 contains an AES 256 hardware accelerator, which could encrypt the payload before passing it to the LoRaWAN module for transmission. The move to AES 256, although not required at time of writing, may become the norm with the increase in viability of quantum computers [30]. Utility companies, in particular, would benefit as they require equipment to be operational for decades.

While an additional component (a phototransistor) is an additional cost it is a low-cost at less than 5% of the cost of the raw materials of the device, and additional software would be required to interact with the sensor device this is a minimal addition, the cost of the change would be shared between all of the required devices. Therefore, the financial considerations are met. The provisioning software only runs immediately after the device was reset and be disabled afterwords. The phototransistor circuit can also be disabled to save power after provisioning; the ability to disable IO pins is inherent in the MSP430 microcontroller and requires no additional hardware. The expected impact on battery life is, therefore, negligible, meeting the low energy considerations.

To test the reliability of the proposed system in different conditions the PCB sensor device was programmed in a laboratory in a variety of conditions from brightly lit during the day to evening conditions with artificial light. In all cases, the system was easily able to distinguish between the LED flash and background light. A feature of the PWM implementation means that the proposed design is both hardware and software agnostic, any mobile phone featuring a flashlight can be reliably used to program devices without having to re-calibrate any equipment, as the light sensor is self-calibrated through measuring the ambient light and then recording any values consistently higher.

As the installation process for these devices is accessible and straightforward, it allows individuals without specialised tools or training to install them, reducing the cost and time associated with creating the network as a whole. Particularly for utility networks, blank LoRaWAN devices could be stored in maintenance vehicles and as maintenance engineers carry out their daily tasks they could also be deploying sensing devices. This would allow for a more rapid deployment and would not require employing specialists to visit various locations. Moreover, if the central key server was compromised, devices could be re-keyed faster, using this larger network of people, than the traditional design of sending each device to the manufacturer. Members of PNDC have evaluated the proposed provisioning mechanism and have acknowledged the significant advantage of not connecting laptops to devices, the provenance of which is often difficult to determine especially when relying on third party contractors.

The proposed design has an apparent limitation where a mobile data connection is required to transfer the matching credentials to the LoRaWAN server. However, in this situation an installer has two options. If they are aware of the lack of mobile coverage they can program the device while approaching the installation site. Alternatively as the LoRaWAN device is programmed to repeatedly attempt to join the LoRaWAN network, the installer can generate the required credentials locally, and transfer the keys to the LoRaWAN device and delay sending of the matching credentials to the LoRaWAN server until IP connectivity resumes. To test for this scenario, the OnePlus 5T was placed into flight mode, and the installation procedure was started. The smartphone successfully transferred the credentials to the LoRaWAN device and once flight mode was disabled the LoRaWAN device successfully joined the network. Next the light was artificially obstructed to intentionally corrupt the message and cause the CRC to fail, resulting in the device resetting and waiting for a valid configuration. At this stage the Android application had not received a confirmation therefore it started transmitting the credentials again to the waiting LoRaWAN device. These programming decisions meant that even without an immediate mobile connection devices were still able to be programmed.

A. Security Analysis

The proposed key transfer system only alters the method for generating and transferring key into the LoRaWAN module. Therefore, this section will explore the benefits and limitations of that method, and the security features of LoRaWAN itself will not be considered. As the credentials are transferred using visible light, it would be possible for an attacker to record the deployment using a high speed video camera to decipher the key. Security in visible light communications is an evolving field [31]. However, due to the contained nature of the proposal, only using a single fixed intensity transmission LED and a receiving diode, this is an on-going research question [32].

Given the predominantly unattended nature of LoRaWAN devices, an attacker could also factory reset a device and attempt to program the device themselves. However, without a corresponding key held on the LoRaWAN server, the join request would fail and prevent the device from joining the network, meaning no malicious data would be injected into the network.

A physical attack could result in a malicious actor obtaining one of the development organisation's smartphones capable of programming devices, and therefore adding new devices to the system. However, this can be prevented by using a passcode or biometrics, a common protection scheme used to secure other systems [33]. Moreover, once such a device is reported missing or tracked as unaccounted for through *find my phone* or similar scheme, it can be blacklisted. Additionally, this would have minimal effect on the existing network as keys only remain on the smartphone until they can be transferred to the LoRaWAN server, and the API on the server only accepts adding keys, with no provision to extract keys.

Another potential attack would be malicious applications running alongside the provisioning application. The installed

keys only remain on the device until they have been securely transferred to the LoRaWAN server, at which point they are removed. Modern mobile applications also benefit from built-in security features such a sandboxing to prevent other applications on the installer's phone from extracting the keys while programming the device. Smartphones also have a well-established set of API for cryptographic operations to ensure sufficient entropy generation for strong key generation [27], and present less of a risk than laptops.

VII. Conclusion

Our system allows non-specialised individuals to easily program low-cost, low-performance devices which commonly rely on symmetric-key cryptography, such a LoRaWAN. The security of these devices depends on the cryptographic keys installed, and therefore should not be configured with default or easily guessable keys. For large scale deployments particularly in critical national infrastructure such as utilities, the cost and security of a deployment is critical. Therefore, two considerations are the cost of deployment and its resilience. The cost of deployment is a factor of requiring internal specialists to program devices, while the resilience is a factor of how quickly the network could be recovered from a compromise. Simplifying the installation process allows the configuration workload to be distributed across various parties, reducing cost and time associated with installation, and allowing the network to be re-keyed faster. The system removes the complexity overhead associated with setting up devices securely, reducing the incentive to deploy devices using default or easily guessable credentials. Moreover, since the system does not require any credentials to be pre-loaded into the devices, the credentials cannot be targeted before installation, meaning devices can be freely distributed without fear of the keys being intercepted. The distributed nature of the installation process also means that if a malicious attacker wished to compromise a large number of devices, they would have to visit each location or influence various employees. As the system uses a smartphone, which is very common and can generate suitably random keys, the additional cost of the hardware on the IoT device is minimal and requires few resources of the micro-controller. Therefore, the system alleviates some of the concerns for deploying sensor networks in real environments as the cost is reduced, while improving security, and if system is compromised it can be re-keyed.

References

[1] M. Eldefrawy, I. Butun, N. Pereira, and M. Gidlund, "Formal security," *Computer Networks*, vol. 148, pp. 328–339, 2019.

[2] N. Hayati, K. Ramli, M. Suryanegara, and Y. Suryanto, "Potential Development of AES 128-bit Key Generation for LoRaWAN Security," in *2019 2nd International Conference on Communication Engineering and Technology (ICCET)*. IEEE, 2019, pp. 57–61.

[3] M. S. Cesar Cerrudo, Esteban Martinez Fayo, "LoRaWAN Networks Susceptible to Hacking: Common Cyber Security Problems, How to Detect and Prevent Them," Ioactive, Tech. Rep., Jan. 2020, accessed: 27 Jun 2020. [Online]. Available: https://act-on.ioactive.com/acton/attachment/34793/f-87b45f5f-f181-44fc-82a8-8e53c501dc4e/1/-/-/-/-/LoRaWAN%20Networks%20Susceptible%20to%20Hacking.pdf

[4] O. Garcia-Morchon, S. Kumar, and M. Sethi, "Internet of Things (IoT) Security: State of the Art and Challenges," Request for comments, Tech. Rep., 2019, accessed 11th November 2019. [Online]. Available: https://rfc-editor.org/rfc/rfc8576.txt

[5] "Implementation of the NIS Directive DfT Guidance version 1.1," Department For Transport, Tech. Rep., Dec 2018, accessed: 13th October 2019. [Online]. Available: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/765786/implementation-of-the-nis-directive-dft-guidance.pdf

[6] I. Butun, N. Pereira, and M. Gidlund, "Security risk and future directions," *Future Internet*, vol. 11, no. 1, p. 3, 2019.

[7] R. McPherson, C. Hay, and J. Irvine, "Using LoRaWAN Technology to Enhance Remote Power Network Monitoring," in *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*, April 2019, pp. 1–5.

[8] Z. Gutterman, B. Pinkas, and T. Reinman, "Analysis of the linux random number generator," in *Proceedings - IEEE Symposium on Security and Privacy*, vol. 1, 2006, pp. 371–385.

[9] J. Liebow-Feeser, "LavaRand in Production: The Nitty-Gritty Technical Details," Cloudflare, Tech. Rep. [Online]. Available: https://blog.cloudflare.com/lavarand-in-production-the-nitty-gritty-technical-details/

[10] S. Tomasin, S. Zulian, and L. Vangelista, "Security Join Procedure for Internet of Things Networks," in *2017 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, Mar. 2017, pp. 1–6.

[11] N. Hayati, K. Ramli, M. Suryanegara, and Y. Suryanto, "Potential Development of AES 128-bit Key Generation for LoRaWAN Security," in *2019 2nd International Conference on Communication Engineering and Technology (ICCET)*, April 2019, pp. 57–61.

[12] J. Han and J. Wang, "An Enhanced Key Management Scheme for LoRaWAN," in *Security, Privacy, and Anonymity in Computation, Communication, and Storage*, G. Wang, J. Chen, and L. T. Yang, Eds. Cham: Springer International Publishing, 2018, pp. 407–416.

[13] J. Zhang, A. Marshall, and L. Hanzo, "Channel-Envelope Differencing Eliminates Secret Key Correlation: LoRa-Based Key Generation in Low Power Wide Area Networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 12, pp. 12 462–12 466, 2018.

[14] W. Xu, S. Jha, and W. Hu, "Exploring the Feasibility of Physical Layer Key Generation for LoRaWAN," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, 2018, pp. 231–236.

[15] H. Bauer, O. Burkacky, and C. Knochenhauer, "Security in the Internet of Things," McKinsey & Company, Tech. Rep., May 2017. [Online]. Available: https://www.mckinsey.com/~/media/McKinsey/Industries/Semiconductors/Our%20Insights/Security%20in%20the%20Internet%20of%20Things/Security-in-the-Internet-of-Things.ashx

[16] R. McPherson and J. Irvine, "Using Smartphones to Enable Low-Cost Secure Consumer IoT Devices," *IEEE Access*, vol. 8, pp. 28 607–28 613, 2020.

[17] "Consumer Internet of Things Security Labelling Survey Research Findings," Department for Digital, Culture, Media and Sport (DCMS), Tech. Rep., 2019. [Online]. Available: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/798543/Harris_Interactive_Consumer_IoT_Security_Labelling_Survey_Report.pdf

[18] E. Leverett, R. Clayton, and R. Anderson, "Standardisation and Certification of the 'Internet of Things'," *Proceedings of WEIS 2017*, 2017.

[19] "Progress of the 2016–2021 national cyber security programme," National Audit Office, Tech. Rep., March 2019. [Online]. Available: https://www.nao.org.uk/wp-content/uploads/2019/03/Progress-of-the-2016-2021-National-Cyber-Security-Programme.pdf

[20] M. Tritschler and W. Mackay, "UK Smart Grid Cyber Security," KEMA, Tech. Rep., June 2011. [Online]. Available: https://www.energynetworks.org/assets/files/electricity/futures/smart_grids/UK_Smart_Grid_Cyber_Security_Report.pdf

[21] E. Barker, "Guideline for using cryptographic standards in the federal government: Cryptographic mechanisms," National Institute of Standards and Technology, Tech. Rep., 2020. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-175Br1.pdf

[22] "Communications Market Report 2019," The Office Of Communications (Ofcom), Tech. Rep., July 2019. [Online]. Available: https://www.ofcom.org.uk/__data/assets/pdf_file/0028/155278/communications-market-report-2019.pdf

[23] "Radio equipment regulations 2017," United Kingdom Department for Business, Energy & Industrial Strategy, Tech. Rep., Dec. 2017. [Online]. Available: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/664824/radio-equipment-regulations-2017-guidance.pdf

[24] J. M. Kahn and J. R. Barry, "Wireless infrared communications," *Proceedings of the IEEE*, vol. 85, no. 2, pp. 265–298, 1997.

[25] J.-h. Choi, E.-b. Cho, T.-G. Kang, and C. G. Lee, "Pulse width modulation based signal format for visible light communications," in *OECC 2010 Technical Digest*. Ieee, 2010, pp. 276–277.
[26] S. Park, D. Kim, and H. Cha, "Reducing energy consumption of alarm-induced wake-ups on android smartphones," in *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, 2015, pp. 33–38.
[27] J. Feichtner, "A comparative study of misapplied crypto in android and ios applications," in *16th International Conference on Security and Cryptography (SECRYPT 2019)*. SciTePress, 2019.
[28] E. Tsimbalo, X. Fafoutis, and R. J. Piechocki, "CRC error correction in IoT applications," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 1, pp. 361–369, 2016.
[29] Opto Semiconductors. Accessed 1st March 2020. [Online]. Available: https://dammedia.osram.info/media/resource/hires/osram-dam-14425800/CW2%20FKAQ81.Z1_EN.pdf
[30] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 1996, pp. 212–219. [Online]. Available: https://arxiv.org/pdf/quant-ph/9605043.pdf
[31] M. A. Arfaoui, M. D. Soltani, I. Tavakkolnia, A. Ghrayeb, M. Safari, C. Assi, and H. Haas, "Physical layer security for visible light communication systems: A survey," *IEEE Communications Surveys & Tutorials*, 2020.
[32] M. A. Arfaoui, A. Ghrayeb, and C. Assi, "Secrecy rate closed-form expressions for the SISO VLC wiretap channel with discrete input signaling," *IEEE Communications Letters*, vol. 22, no. 7, pp. 1382–1385, 2018.
[33] L. Sharma and M. Mathuria, "Mobile banking transaction using fingerprint authentication," in *2018 2nd International Conference on Inventive Systems and Control (ICISC)*, 2018, pp. 1300–1305.

ROSS MCPHERSON Received the B.Eng. (Hons.) degree in Computer & Electronic Systems from University of Strathclyde, Glasgow, UK, in 2016. He is currently pursuing a Ph.D. degree in cyber security in the internet of things devices at the University of Strathclyde. Focusing on the practical development and usage of devices without detracting from the existing user experience.

He has previously worked in the financial, utility and mobile communication industries. During his time studying he has created private mobile and low-power wide area communication networks, and retrofitted existing hardware to enable secure remote communication.

JAMES IRVINE currently leads the communications and system integration theme of the Power Networks Demonstration Centre, part of the EEE Department at the University of Strathclyde in Glasgow, from where he obtained his PhD in 1994. His research focusses on mobile communication and security, in particular on resource allocation and coding theory.

He was active for many years in the UK Mobile VCE research programme, as Academic Co-ordinator of the 'Instant Knowledge' work area on privacy and trust in new personal communication services, and he led security work in the previous two MVCE Core programmes. His conference activities include General chair of VTC2015-Spring, TPC Chair of VTC2004-Spring, and TPC Co-chair of ISWCS2007. He is a co-author of two books and over 140 journal and conference papers. A co-author of seven patents, he has testified in the UK and the Netherlands on multiple cases involving cellular radio technology.

Dr Irvine is a member of the IEEE Blockchain Initiative Steering Group, chairing the events activity. He is an elected Board member of the IEEE Vehicular Technology Society, and was President of the Society 2008-9 and founding Editor in Chief of the IEEE Vehicular Technology Magazine.